GARUDA Storage Resource Manager (GSRM)
Version 1.0
User Manual

16th June 2010

System Software Development Group
C-DAC KP
Bangalore

GARUDA Storage Resource Manager

Version 1.0

User Manual

Project No: CDAC/B/SSDG/GDS/2010/033		
Document No: SSDG/GDS/2010/User manual/ 1.0		
Control Status: Controlled (Internal Circulation Only)		
Author : Shashidhar.V and Payal Saluja		
Distribution List : Project 1. GDS Team 2. Dr B B Prahlada Rao 3. Dr Subrata 4. GARUDA application Users		
Approved By Name Signature	**Designation**	**Date of Approval**
Dr. B B Prahlada Rao	Program Co-ordinator,SSDG, CDAC, Bangalore	
Release By : SSDG, C-DAC Knowledge Park, Bangalore		
Date of Release : 16/06/2010		
Copy No. : 1		

Contents Page No.

1 GSRM Introduction 4-7
1.1. GSRM Overview……4
1.2. Purpose and Scope of the Document……4
1.3. GSRM Highlights & Supported features……4
1.4. GSRM Test Bed Deployment……5
1.5. GSRM Accessibility……6
1.6. GSRM Client Interfaces……7

2 GSRM Interface Usage 8-27
2.1 GSRM Access Points……7
2.2 Pre-requisites for using GSRM……7
2.3 SToRM command line……8
2.4 DPM Web Interface……12
2.5 DPM C APIs……13
2.6 BestMan command line……18
2.7 BestMan Java APIs……22

3 Support……25
4 Glossary……26

1. GSRM Introduction

1.1. Overview:

Data grid is a system that manages and controls large amounts of distributed data. Heterogeneous Grid environments demand a global namespace in order to present a Uniform storage space, providing transparency of location as well as access to data. Grid applications interact with a variety of storage systems. Hence, there is a need to standardize and streamline the uniform access interface to grid storage system. Such a system not only manages storage allocation in a transparent manner, but also dynamically manages the spaces on Grid.

GARUDA Storage Resource Manager is a peer to peer data grid solution for SOA based GARUDA. **An SRM can be defined as a middle ware component** that manages the dynamic use of storage resources on the Grid. This means that space can be allocated dynamically to a client, and that the decision of which files to keep in the storage space is controlled dynamically by the SRM. GSRM is a disk based SRM implementation, following SRM version 2.2 specifications (https://sdm.lbl.gov/srm-wg/). It adheres to Open Grid Forum standards. It is based on open source Disk pool Manager (DPM) Version 1.7.2

1.2. Purpose and Scope of the Document

This document is intended for the GARUDA users. The purpose of this document is to help GARUDA users to use GSRM storage services. It describes the step by step procedure to access GARUDA storage resources using different SRM client interfaces.

1.3. Highlights and Features:

The GSRM version1.0 supports the following features:

- Provides single access point to distributed and heterogeneous storage resources for users by providing **global namespace**
  - Provides **File and Directory managemen**t--
- Supports GSI, Access Control Lists, and VOMS **Security mechanism** and supports secure data transfers.
- GSRM provides dynamic **space management** reserving & release of storage spaces and setting lifetime for the files.
- User **Quota Allocation**: Each user is allocated 10GB of default, which can be increased based on the user request.
- Provides **user friendly interfaces** to access its services.
- Provides **interoperability** with other SRM implementations, based on SRMv2.2 specifications like Best Man – Berkeley Storage Manager, Castor-SRM, dCache-SRM, StoRM - Storage Resource Manager. Users can access GSRM storage services using any of the above mentioned client interfaces.

Figure 1 shows the various GSRM components and their feature mapping. GSRM components are depicted with green boxes and their corresponding functionality with orange boxes.

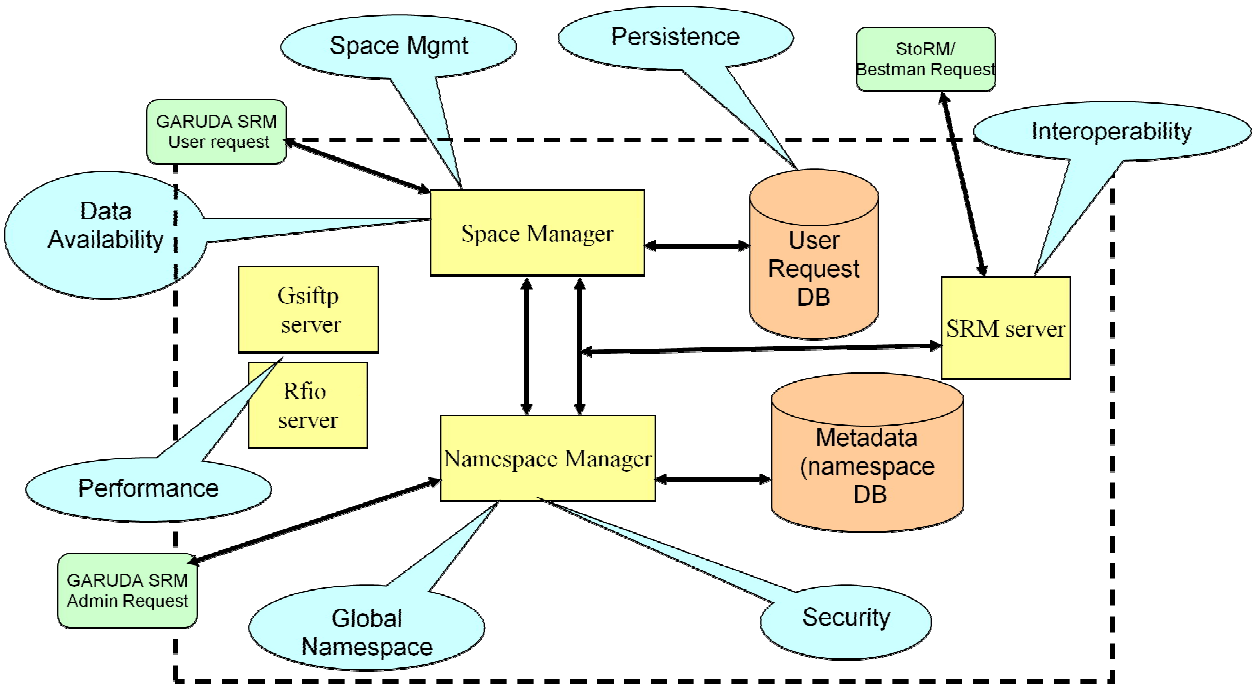


Figure 1 GSRM Component-Features Mapping

1.4. GSRM Test Bed Deployment:

Figure 2 shows the GSRM deployment scenario across four GARUDA centers: CDAC Bangalore, CDAC Pune, CDAC Chennai and CDAC Hyderabad with approximate storage of 1.2TB.

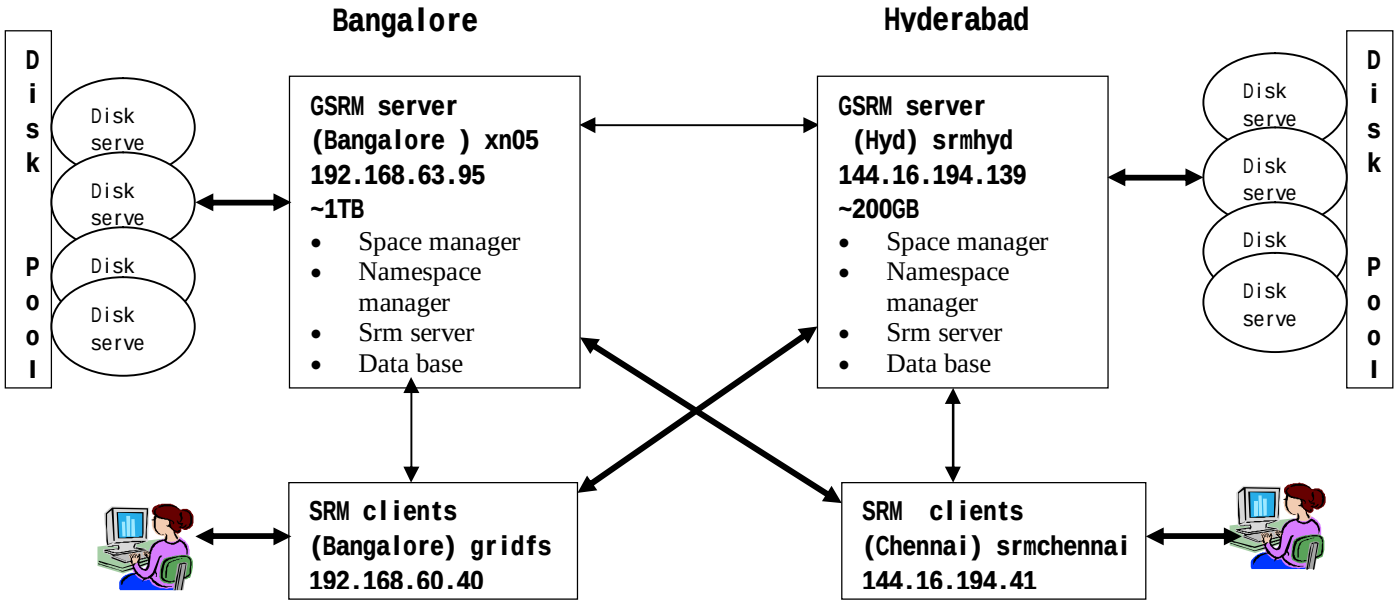


Figure 2 GSRM Deployment

Above shown is the current test bed setup with servers being installed Bangalore (**192.168.63.95)**

and Hyderabad (**144.16.194.139)** and clients installations at Bangalore (**gridfs-192.168.60.40**), and Chennai(**144.16.194.41).** Users can access GSRM services through **gridfs(192.168.60.40).** We also provide users, a test node **pvfs2 (172.20.1.81)** at Bangalore where all SRM clients has been installed and users can practice using SRM client interfaces. The three daemons running at each of the GSRM servers address the requests coming from any of the SRM client installations**. Space Manager** is the main daemon performing space allocation, space release, and cleanup. It also keeps track of each and every user request received from SRM server. **Namespace Manager** takes care of the security of the system by controlling the hierarchical namespace, the file permissions and the mapping between SFN (Site File Name) and physical names.**SRM server** convert the SRM requests into GARUDA-SRM requests and distribute the SRM requests workload to the required component. **Disk Servers**: provides secure file transfers between the disks and the client. **Request database** stores the metadata about each user data request and **Namespace database** stores the mapping between physical file name and logical file name. **Disk Pools** are the collection of various disks or file systems part of GSRM system. These are the virtual entities where the actual data gets stored.

1.5. GSRM Accessibility in GARUDA:

GARUDA services are available to users at GARUDA head node(currently only at Bangalore—gridfs**(192.168.60.40)**).The storage systems supported in this version of GSRM(v1.0) are disk based storage. Users can either access GSRM storage through command line clients installed at gridfs . So, users can download the input files from GSRM before submitting the job and can also store the generated output files in it after the completion of the job. GSRM will also accessible through GARUDA portal in future. Figure 3 depicts the GSRM integration and interaction with other GARUDA middleware components with respect to the job submission by the user.

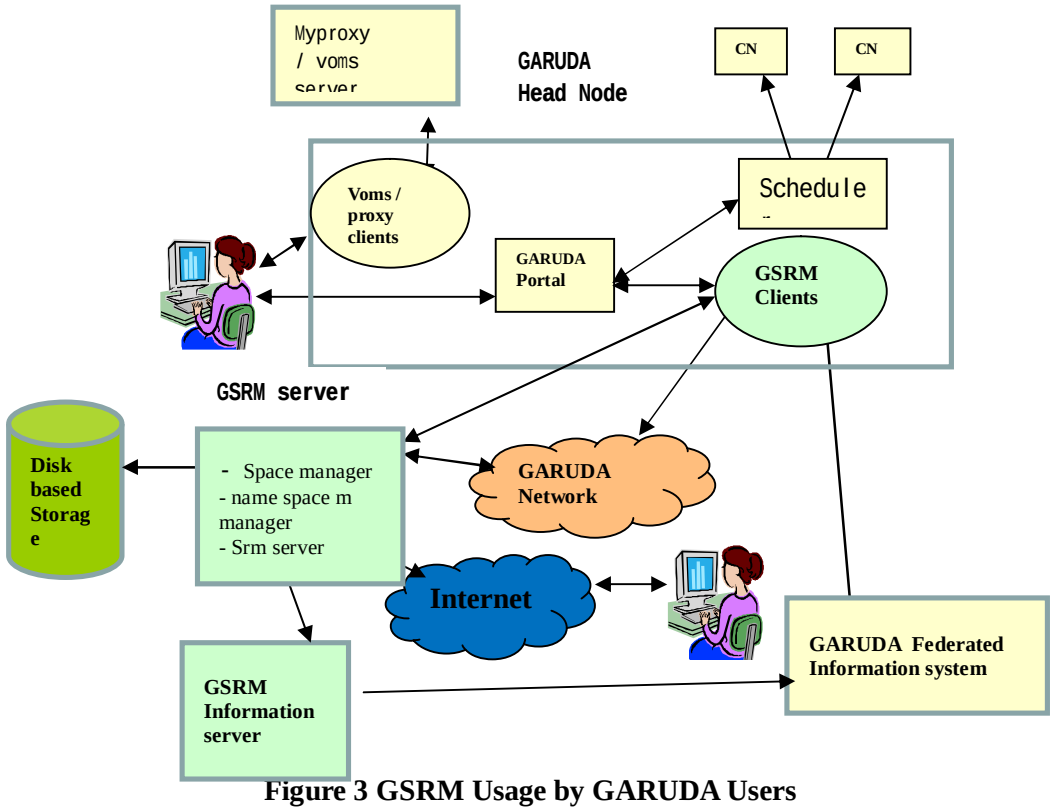


Figure 3 GSRM Usage by GARUDA Users

Steps to be followed to access GSRM from gridfs:

1. Login to gridfs(192.168.60.40)
2. Upload your IGCA user certificates under /<user-home-dir>./.globus
3. Initialize proxy with grid-proxy-init
4. Set environmental variables mentioned in section, respectively for whichever client to be used.
5. Run the SRM commands (Details discussed in section 2 for each interface).

1.6 GSRM Client Interfaces:

GSRM services can be accessed by the following client interfaces:

i. **Command Line :**The various command line interfaces available to access GSRM services are StoRM (clientSRM commands), BestMan and dpm command line interfaces

ii. **APIs:** GSRM supports DPM 'C' APIs for C based applications and BestMan 'Java' APIs for Java based applications..

iii. **Web Interface:** DPM web interface is available to access GSRM services through browser.

2. GSRM Interface Usage

2.1 GSRM Access points:

Currently, GSRM services can be accessed logging into GARUDA Bangalore head node **gridfs (192.168.60.40)** or **pvfs2 (172.20.1.81). SRM c**lient installations are available on both the above mentioned nodes.

1. **pvfs2(172.20.1.81)** node should be used to just test all the available SRM client interfaces like StoRM,DPm,BestMan.
2. **gridfs (192.168.60.40)** node should, if the user wishes to use GSRM storage for job execution. Users can download /Upload , input/output files into GSRM while submitting jobs from gridfs.

2.2 Pre-requisites for using the GSRM services :

- A valid user IGCA certificate (Valid user certificate can be obtained through IGCA web site :: https://ca.garudaindia.in/
- A SRM endpoint to contact **(httpg://xn05.ctsf.cdac.org.in:8446**)
- Unix Login for GARUDA Grid Head node **gridfs (mail to :grid-help@cdacb.ernet.in)** or **pvfs2 (mail to :** rt-gds@cdacb.ernet.in**)**
- Authorization to access SRM storage resources.(mail to : rt-gds@cdacb.ernet.in**)**
- SRM client interface Installation **(Any of the above mentioned SRM client installation)**

- Following **Access mechanisms** are available at above mentioned nodes to access GSRM:

1. **gridfs(192.168.60.40) :** gridfs is the Bangalore GARUDA head node. GSRM services can be accessed from here using **StoRM command line** interface.

If the user wants to use the clientSRM (StoRM Clients) from gridfs machine

1. Create a valid user proxy
grid-proxy-init
Enter GRID pass phrase for this identity:
Creating proxy..... Done
Your proxy is valid until: <Date/ Time>

2. Set the env variable for Globus location path
export
GLOBUS_LOCATION=$GLOBUS_LOCATION:/usr/local/GARUDA/GLOBUS-4.0.7/
export PATH=$PATH:/opt/gsrm-client/srmv2storm/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/gsrm- client/cgsi_soap/lib

3. Run the clientSRM command

2. **pvfs2 (172.20.1.81):** pvfs2 is the GSRM testing node with the following client interfaces installed
   - StoRM, Bestman and dpm command line interfaces
   - Bestman Java APIs
   - DPM C APIs

   The usage of above mentioned interfaces has been discussed on the following sub-sections.

3. **GSRM Web Client** is accessible from any of the user machines reachable to GSRM server (xn05.ctsf.cdac.org.in), using URL -- **https://xn05.ctsf.cdac.org.in/**

2.3 StoRM command line usage:

The usage of StoRM command line client is as follows:

clientSRM -h
Usage: clientSRM <requestName> <requestOptions> <requestName>

The various functions supported by StoRM are :

i. Discovery functions:
   Ping, GetTransferProtocols (or gtp)
ii. Directory functions:
   Mkdir, Rmdir, Rm, Ls, Mv
iii. Data transfer functions:
   PtP, PtG
   Copy(or cp), StatusPtP(or sptp), StatusPtG (or sptg)
   StatusCopy(or scp), PutDone(or pd),

ReleaseFiles (or rf), BringOnline (or bol)
StatusBringOnline (or sbol)
GetRequestTokens(or grt)
GetRequestSummary(or grs)
AbortRequest(or ar)
AbortFiles(or af)
ExtendFileLifeTime (or efl)

iv. Space management functions:

ReserveSpace (or rs)
GetSpaceMetadata (or gsm)
ReleaseSpace (or rsp)
GetSpaceTokens (or gst)

Type "clientSRM <requestName> -h" to get help about <requestName> options
Global options:
-h, --help Print this help message

-v, --version Print version information

Example 1: Ping to GSRM server "xn05.ctsf.cdac.org.in"

Command Usage:
clientSRM ping -e httpg://xn05.ctsf.cdac.org.in:8446/

Output :

```
============================================================
Sending Ping request to: httpg://xn05.ctsf.cdac.org.in:8446/
============================================================
Request status:

 statusCode="SRM_SUCCESS"(0)
 explanation="SRM server successfully contacted"
============================================================

SRM Response:
 versionInfo="v2.2"
 otherInfo (size=2)
  [0] key="backend_type"
  [0] value="DPM"
  [1] key="backend_version"
  [1] value="1.7.2-4"
```

Example 2: Upload a file in GSRM storage
It involves 3 steps
step 1: ptp (prepare-to-put request) request to GSRM server
step 2: Invoke the file transfer
step 3: Ensure the file has transfered i.e., pd (putdone)

step 1: ptp request

Command:

clientSRM ptp -e httpg://xn05:8446/ -s srm://xn05/dpm/ctsf.cdac.org.in/home/test/fastainputfile -p

Output:

```
============================================================
Sending PtP request to: httpg://xn05:8446/
============================================================
Polling request status:
Current status: SRM_REQUEST_QUEUED (Ctrl+c to stop polling).
============================================================
Request status:
  statusCode="SRM_SUCCESS"(0)
============================================================
SRM Response:
  requestToken="635388bb-2e92-4a9f-b590-57c99dfcdb38"
  arrayOfFileStatuses (size=1)
     [0] SURL="srm://xn05/dpm/ctsf.cdac.org.in/home/test/fastainputfile"
     [0] status: statusCode="SRM_SPACE_AVAILABLE"(24)
     [0] fileSize=0
     [0] remainingPinLifetime=7199
     [0] remainingFileLifetime=0
```

TURL="gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/data01/test/2010-05-05/fastainputfile.1312.0"

==========

Step 2: Globus File transfer (File actually gets transferred to SRM location)

globus-url-copy <source> < target >

i.e., globus-url-copy source TURL(returned by step 1)

Command :

globus-url-copy file:/home/shashi/em_rel_std_pln gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/data01/test/2010-05-05/fastainputfile.1312.0

Step 3: pd (putdone) – use the token returned in step 1

Command:

clientSRM pd -e httpg://xn05:8446/ -s srm://xn05/dpm/ctsf.cdac.org.in/home/test/fastainputfile -t 635388bb-2e92-4a9f-b590-57c99dfcdb38

Output:

```
============================================================
Sending PutDone request to: httpg://xn05:8446/
============================================================
Request status:
  statusCode="SRM_SUCCESS"(0)
```

```
============================================================
SRM Response:
  arrayOfFileStatuses (size=1)
     [0] SURL="srm://xn05/dpm/ctsf.cdac.org.in/home/test/fastainputfile"
     [0] status: statusCode="SRM_SUCCESS"(0)
============================================================
```

Example 3: Download a file from GSRM storage

Step 1: ptg (prepare-to-get) request to GSRM server

Command :

clientSRM ptg -e httpg://xn05:8446/ -s srm://xn05/dpm/ctsf.cdac.org.in/home/test/testfile2 -p

Output:

```
============================================================
Sending PtG request to: httpg://xn05:8446/
============================================================
Polling request status:
Current status: SRM_REQUEST_QUEUED (Ctrl+c to stop polling).
============================================================
Request status:
  statusCode="SRM_SUCCESS"(0)
============================================================
SRM Response:
  requestToken="8003ef88-5624-4710-970a-5498671ad08c"
  arrayOfFileStatuses (size=1)
     [0] status: statusCode="SRM_FILE_PINNED"(22)
     [0] sourceSURL="srm://xn05/dpm/ctsf.cdac.org.in/home/test/testfile2"
     [0] fileSize=5990933
     [0] remainingPinTime=7199
     [0]
transferURL="gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/
data01/test/2010-01-09/testfile2.771.0"
```

Step 2: File transfer

Command:

globus-url-copy gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/data01/test/2010-01-09/testfile2.771.0 file:/home/shashi/get_testfile2

2.4 DPM Web Interface Usage:

DPM Web interface provides user friendly GUI to access GSRM services It can be accessed using the url : https://<dpm-head-node>

Test Bed URL : https://xn05.ctsf.cdac.org.in

DPM provides a basic GUI with minimal functions like file & directory management , file upload/download etc.

Steps to Access DPM Web interface :

1. Upload the user certificate (pkcs12 format) file into the browser(refer link : http://ca.garudaindia.in/index.php/information/faq/

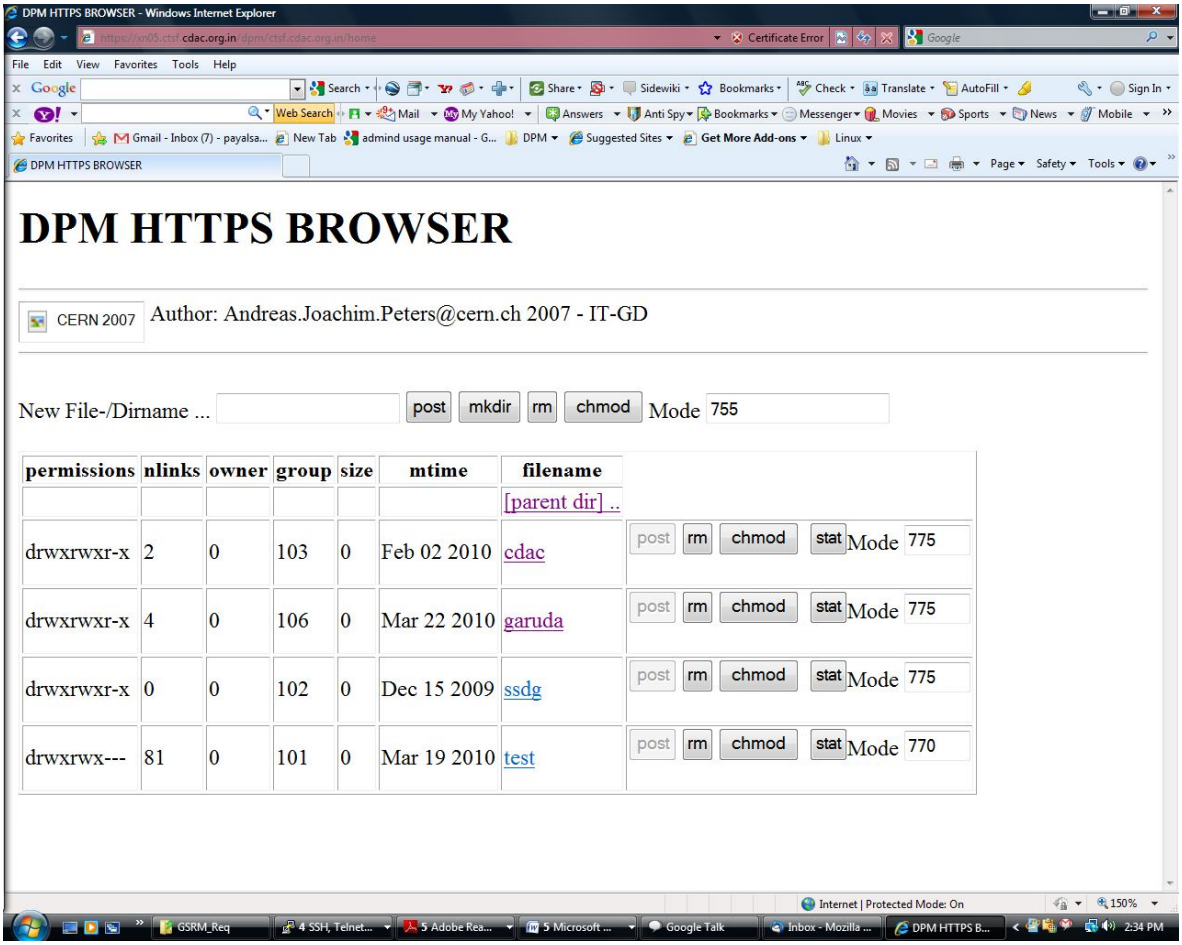
2. https://<dpm-head-node>

3.

Note:

1. There is a bug (requires selection of file to be uploaded within 10 seconds) in the file upload option of the DPM web interface

2. A sample screenshot of the DPM https browser is shown below


DPM HTTPS BROWSER - Windows Internet Explorer
DPM HTTPS BROWSER
CERN 2007
Author: Andreas.Joachim.Peters@cern.ch 2007 - IT-GD
New File-/Dirname ...
post
mkdir
rm
chmod
Mode 755
permissions nlinks owner group size mtime filename
[parent dir] ..
drwxrwxr-x 2 0 103 0 Feb 02 2010 cdac
drwxrwxr-x 4 0 106 0 Mar 22 2010 garuda
drwxrwxr-x 0 0 102 0 Dec 15 2009 ssdg
drwxrwx--- 81 0 101 0 Mar 19 2010 test
stat
Mode 775
Mode 770

2.5 DPM C APIs Usage :

ref url: https://twiki.cern.ch/twiki/bin/view/LCG/DpmApi

Table 1 shows the list of DPM 'C' APIs currently supported by GSRM

COMMAND	DESCRIPTION
dpm_copy	copy a set of existing files
dpm_get	make a set of existing files available for I/O
dpm_getstatus_copyreq	get status for a dpm_copy request
dpm_getstatus_getreq	get status for a dpm_get request
dpm_getstatus_putreq	get status for a dpm_put request
dpm_put	make a set of existing files available for I/O
dpm_putdone	mark a set of files as complete

Table 1 List of DPM

Example 1 : A sample C Program using DPM C APIs to Show the supported transfer protocols by the GSRM.

Code :

```
#include <sys/types.h>
#include "/opt/lcg/include/dpm/dpm_api.h"
#include "/opt/lcg/include/dpm/serrno.h"

main()
{
    int i;
    int nb_supported_protocols;
    char **supported_protocols;

    if (dpm_getprotocols (&nb_supported_protocols, &supported_protocols) < 0) {
        sperror ("dpm_getprotocols");
        exit (2);
    }
    printf ("Supported protocols are:");
    for (i = 0; i < nb_supported_protocols; i++) {
        printf (" %s", supported_protocols[i]);
        free (supported_protocols[i]);
    }
    printf ("\n");
    free (supported_protocols);
    exit (0);
}
```

Execution:

export DPM_HOST=xn05.ctsf.cdac.org.in
export DPNS_HOST=xn05.ctsf.cdac.org.in
gcc –o testprotocols dpm_testgetprotocols.c –I /opt/lcg/include/dpm/ -L /opt/lcg/lib/

Results:
./testprotocols
Supported protocols are: rfio gsiftp https

Example 2: A sample C Program using DPM C APIs to Upload a File into GSRM Storage

Code :

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include “dpm_api.h”
#include “serrno.h”
#define DEFPOLLINT 10
#include “globus_common.h”
#include “globus_ftp_client.h”

globus_mutex_t lock;
globus_cond_t cond;
globus_bool_t done;

main(argc, argv)
int argc;
char **argv;
{
      static char *f_stat[] = {“Success”, “Queued”, “Active”, “Ready”, “Running”, “Done”, “Failed”, “Aborted”};
      struct dpm_putfilestatus *filestatuses;
      struct dpm_filestatus *pdfilestatuses;
      int i, j;
      int nbfiles;
      int nbprotocols;
      int nbreplies, nbpdreplies;
      static char *protocols[] = {“gsiftp”};
      int r = 0;
      char **surls;
      char r_token[CA_MAXDPMTOKENLEN+1];
      struct dpm_putfilereq *reqfiles;
      int status;
      char *source_array[10];
      char *dest_array[10];
```

```
        char *surl_array[10];
        globus_ftp_client_handle_t *handle ;
        globus_ftp_client_handleattr_t handle_attr ;
        int thirdparty_file_transfer_cb(void* ,globus_ftp_client_handle_t* ,globus_object_t*
);
        const char *source_url;
        globus_ftp_client_operationattr_t *source_attr, *dest_attr, *attr;
        const char *dest_url;
        globus_object_t *error;

              if (argc < 2) {
               fprintf (stderr, "usage: %s SURLs SOURCE_TURLs \n", argv[0]);
              exit (1);
}
         if(argc/2 != 0)
        nbfiles = argc/2 ;
        if ((reqfiles = calloc (nbfiles, sizeof(struct dpm_putfilereq))) == NULL) {
              perror ("calloc");
               exit (1);
        }
        for (i = 0; i < nbfiles; i++)
              reqfiles[i].to_surl = argv[i+1];
        nbprotocols = sizeof(protocols) / sizeof(char *);
        if ((status = dpm_put (nbfiles, reqfiles, nbprotocols, protocols, NULL,
           0, 0, r_token, &nbreplies, &filestatuses)) < 0) {
               sperror ("dpm_put");
               exit (1);
        }

        printf ("dpm_put returned r_token: %s\n", r_token);
        /* wait for request status "Done" or "Failed" */
        while (status == DPM_QUEUED || status == DPM_ACTIVE) {
              for (i = 0; i < nbreplies; i++) {
                    if ((filestatuses+i)->to_surl)
                          free ((filestatuses+i)->to_surl);
                    if ((filestatuses+i)->turl)
                          free ((filestatuses+i)->turl);
                    if ((filestatuses+i)->errstring)
                          free ((filestatuses+i)->errstring);
              }
              free (filestatuses);
             sleep ((r++ == 0) ? 1 : DEFPOLLINT);
              if ((status = dpm_getstatus_putreq (r_token, 0, NULL,
                 &nbreplies, &filestatuses)) < 0) {
                    sperror ("dpm_getstatus_putreq");
                    exit (1);
```

```
            }
        }
        printf ("request state %s\n", status == DPM_DONE ? "Done" : "Failed");
        if (status == DPM_FAILED)
            exit (1);
    for (i = 0; i < nbreplies; i++) {
           if ((filestatuses+i)->turl)
           {
            dest_array[i]=(char *)malloc(strlen((filestatuses+i)->turl));
                if(dest_array[i] == NULL)

                printf(" malloc error \n");
               strcpy(dest_array[i],(filestatuses+i)->turl);
               printf ("state[%d] = %s, TURL = %s\n", i,
                   f_stat[(filestatuses+i)->status >> 12],
                   (filestatuses+i)->turl);
           }
           else if (((filestatuses+i)->status & DPM_FAILED) == DPM_FAILED)
                printf ("state[%d] = %s, serrno = %d, errstring = <%s>\n", i,
                   f_stat[(filestatuses+i)->status >> 12],
                  (filestatuses+i)->status & 0xFFF,
                  (filestatuses+i)->errstring ? (filestatuses+i)->errstring : "");
           else
                printf ("state[%d] = %s\n", i,
                   f_stat[(filestatuses+i)->status >> 12]);
           if ((filestatuses+i)->to_surl)
                free ((filestatuses+i)->to_surl);
           if ((filestatuses+i)->turl)
                free ((filestatuses+i)->turl);
           if ((filestatuses+i)->errstring)
               free ((filestatuses+i)->errstring);
        }
        free (filestatuses);
for(i=nbfiles+1,j=0;i<argc;i++)
{
printf(" nfiles = %d",nbfiles);
source_array[j++]=argv[i];
printf(" source_array[%d] = %s \n",j-1,source_array[j-1]);
}
for(i=0;i<nbfiles;i++)
surl_array[i]=argv[i+1];
// ##################################### globus C apis for third party transfer
#####################################################
 if(globus_module_activate(GLOBUS_FTP_CLIENT_MODULE)                    !=
GLOBUS_SUCCESS)
 printf(" failed to load the globus ftp module \n");
```

```
 if(globus_ftp_client_handleattr_init(&handle_attr) != GLOBUS_SUCCESS)
   printf(" client handleattr init failed \n");
 if(globus_ftp_client_operationattr_init(&attr) != GLOBUS_SUCCESS)
   printf(" operation attr init failed\n");
 if(globus_ftp_client_handle_init(&handle,&handle_attr) != GLOBUS_SUCCESS)
   printf(" client handle init failed \n");
 for(i=0;i<nbfiles;i++)
 {
   done=GLOBUS_FALSE;
   if(globus_ftp_client_third_party_transfer(&handle,
source_array[i],GLOBUS_NULL, dest_array[i], GLOBUS_NULL ,GLOBUS_NULL,
thirdparty_file_transfer_cb, GLOBUS_NULL)!=GLOBUS_SUCCESS)

   printf(" globus ftp client third party transfer failed \n");
 else
   printf(" success \n");
globus_mutex_lock(&lock);
   while (!done) {
       globus_cond_wait(&cond, &lock);
   }
globus_mutex_unlock(&lock);
}
if(globus_module_deactivate_all()!=GLOBUS_SUCCESS)
  printf("Failed to  deactivate the globus ftp module \n");
//################################### end of globus C apis for 3rd party transfer
#################################################################
for(i=0;i<nbfiles;i++)
{
free(dest_array[i]);
}
//################################### check for put done
surls=surl_array;
     if ((status = dpm_putdone (r_token, nbfiles, surls, &nbpdreplies,
          &pdfilestatuses)) < 0) {
             sperror ("dpm_putdone");
             exit (1);
       }
       for (i = 0; i < nbreplies; i++) {
             if (((pdfilestatuses+i)->status & DPM_FAILED) == DPM_FAILED)
                   printf ("state[%d] = %s, serrno = %d, errstring = <%s>\n", i,
                       f_stat[(pdfilestatuses+i)->status >> 12],
                       (pdfilestatuses+i)->status & 0xFFF,
                      (pdfilestatuses+i)->errstring ? (pdfilestatuses+i)->errstring : "");
             else
                    printf ("state[%d] = %s\n", i,
                       f_stat[(pdfilestatuses+i)->status >> 12]);
```

```
            if ((pdfilestatuses+i)->surl)
                free ((pdfilestatuses+i)->surl);
            if ((pdfilestatuses+i)->errstring)
                free ((pdfilestatuses+i)->errstring);
        }
       free (pdfilestatuses);
//################################## end of put done       exit (0);
}
 int       thirdparty_file_transfer_cb(void*       user_arg,       globus_ftp_client_handle_t*
handle,globus_object_t*  err) {
   char *tmp;
   if (err != GLOBUS_SUCCESS) {
      tmp = globus_object_printable_to_string(err);
      globus_libc_fprintf(stderr,"\n\t\t Error in callback : %s\n",tmp);
   }
   else {
      globus_libc_printf(" \n\t\t Status : file transfer successful \n");
   }
   globus_mutex_lock(&lock);
   done=GLOBUS_TRUE;
   globus_cond_signal(&cond);
   globus_mutex_unlock(&lock);
   return 0;
}
```

Compilation :

gcc –g –o dpm_testput2 dpm_testput2.c –I /opt/lcg/include/dpm/ -I /opt/globus-4.0.7/include/gcc32dbgpthr/ -L /opt/lcg/lib/ -ldpm –ldl –L /opt/globus-4.0.7/lib/ -lglobus_ftp_client_gcc32dbgpthr

Execution:

usage: ./dpm_testput2 SURLs SOURCE_TURLs

./dpm_testput2 srm://xn05/dpm/ctsf.cdac.org.in/home/test/capi_putfile4 gsiftp://g11/home/shashi/env.srm

Result :

dpm_put returned r_token: e5a10f76-2a28-4742-8d64-55a0f6135298
request state Pending
request state Failed
state[0] = Ready, TURL = gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/data01/test/2010-05-05/capi_putfile4.1324.0
nfiles = 1 source_array[0] = gsiftp://g11/home/shashi/env.srm
globus c api
success

Status : file transfer successful

state[0] = Done

Example 2 : To Download a File from GSRM Storage
Its is similar to put a file to GSRM server …
APIS to be used are
dpm_get()
dpm_getstatus_getreq()
globus_ftp_client_third_party_transfer()

NOTE : All the above sample programs are available on "pvfs2" node
NOTE: Refer the list of available apis of DPM and DPNS and also the corresponding manpages

ref url : https://twiki.cern.ch/twiki/bin/view/LCG/DpmApi
http://grid-deployment.web.cern.ch/grid-deployment/documentation/LFC_DPM/dpm/

2.6 BestMan Command line Usage:

Example 1: ping to gsrm server "xn05"

Command Usage:

srm-ping –serviceurl
httpg://xn05.ctsf.cdac.org.in:8446/dpm/ctsf.cdac.org.in/home/garuda

Output:

```
srm-ping   2.2.1.3.6   Mon Nov 30 09:10:48 PST 2009

SRM-Clients and BeStMan Copyright(c) 2007-2009,
Lawrence Berkeley National Laboratory. All rights reserved.
Support at SRM@LBL.GOV and documents at http://datagrid.lbl.gov/bestman
SRM-CLIENT: Connecting to service url
httpg://xn05.ctsf.cdac.org.in:8446/dpm/ctsf.cdac.org.in/home/garuda
SRM-PING: Wed Feb 10 14:44:23 IST 2010 Calling SrmPing Request…
versionInfo=v2.2
Extra information (Key=Value)
backend_type=DPM
backend_version=1.7.2-4
```

Example 2 : Upload a file into GSRM server

Uploading a file can be done using "srm-copy" command
Usage : srm-copy srcurl targeturl -serviceurl srm://gsrmserver/../../

Output :

srm-copy file:/tmp/file1 srm://xn05/dpm/ctsf.cdac.org.in/home/test/inputfile1 -serviceurl srm://xn05:8446/ dpm/ctsf.cdac.org.in/home/test

srm-copy 2.2.1.3.6 Mon Nov 30 09:10:48 PST 2009
SRM-Clients and BeStMan Copyright(c) 2007-2009,
Lawrence Berkeley National Laboratory. All rights reserved.
Support at SRM@LBL.GOV and documents at http://datagrid.lbl.gov/bestman
SRM-CLIENT: Thu May 06 13:56:58 IST 2010 Connecting to httpg://xn05:8446/dpm
SRM-CLIENT: Thu May 06 13:56:59 IST 2010 Calling SrmPrepareToPutRequest now …
request.token=200850ef-2def-40ab-bebe-da3d0107e11d
Request.status=SRM_REQUEST_QUEUED
explanation=null
SRM-CLIENT: Next status call in 10 seconds.
SRM-CLIENT: Thu May 06 13:57:10 IST 2010 Calling Status at Thu May 06 13:57:10 IST 2010
SRM-CLIENT: Result Status from SRM (srmStatusOfPutRequest)=SRM_SUCCESS
SRM-CLIENT: RemainingPinTime=7190
SRM-CLIENT: RemainingFileLifeTime=0
SRM-CLIENT: received
TURL=gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/data01/test/2010-05-06/inputfile1.1331.0

SRM-CLIENT: Thu May 06 13:57:10 IST 2010 start file transfer
SRM-CLIENT:Source=file:////tmp/file1
SRM-CLIENT:Target=gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/data01/test/2010-05-06/inputfile1.1331.0
SRM-CLIENT: Thu May 06 13:57:12 IST 2010 end file transfer for file:/tmp/file1
SRM-CLIENT: Thu May 06 13:57:12 IST 2010 Calling putDone for srm://xn05/dpm/ctsf.cdac.org.in/home/test/inputfile1
Result.status=SRM_SUCCESS

SRM-CLIENT: Request completed with success
SRM-CLIENT: Printing text report now …
SRM-CLIENT*REQUESTTYPE=**put**
SRM-CLIENT*TOTALFILES=1
SRM-CLIENT*TOTAL_SUCCESS=1
SRM-CLIENT*TOTAL_FAILED=0
SRM-CLIENT*REQUEST_TOKEN=200850ef-2def-40ab-bebe-da3d0107e11d
SRM-CLIENT*REQUEST_STATUS=SRM_SUCCESS
SRM-CLIENT*SOURCEURL[0]=file:/tmp/file1
SRM-CLIENT*TARGETURL[0]=srm://xn05/dpm/ctsf.cdac.org.in/home/test/inputfile1
SRM-CLIENT*TRANSFERURL[0]=gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/data01/test/2010-05-06/inputfile1.1331.0
SRM-CLIENT*ACTUALSIZE[0]=0
SRM-CLIENT*FILE_STATUS[0]=SRM_SUCCESS
SRM-CLIENT*EXPLANATION[0]=SRM-CLIENT: PutDone is called successfully

Example 3: Download a file from GSRM storage

Downloading a file can be done using “srm-copy” command

Usage : srm-copy srcurl targeturl -serviceurl srm://gsrmserver/../../

srm-copy srm://xn05/dpm/ctsf.cdac.org.in/home/test/inputfile1 file:/tmp/getfile1 -serviceurl srm://xn05:8446/dpm

Output :

```
srm-copy   2.2.1.3.6   Mon Nov 30 09:10:48 PST 2009

SRM-Clients and BeStMan Copyright(c) 2007-2009,
Lawrence  Berkeley National Laboratory. All rights reserved.
Support at SRM@LBL.GOV and documents at http://datagrid.lbl.gov/bestman
SRM-CLIENT: Thu May 06 13:58:09 IST 2010 Connecting to
httpg://xn05:8446/dpm
SRM-CLIENT: Thu May 06 13:58:10 IST 2010 Calling SrmPrepareToGet Request
now

request.token=686b6b39-81bf-497f-a645-65e7f11b69ae
Request.status=SRM_REQUEST_QUEUED
Request.explanation=null
SRM-CLIENT: Next status call in 10 seconds.
SRM-CLIENT: Thu May 06 13:58:21 IST 2010 Calling Status at Thu May 06
13:58:21 IST 2010
Request.status=SRM_SUCCESS

SRM-CLIENT: RemainingPinTime=7190
SRM-CLIENT: SURL=srm://xn05/dpm/ctsf.cdac.org.in/home/test/inputfile1
SRM-CLIENT: FileStatus code from server=SRM_FILE_PINNED
SRM-CLIENT: Explanation from server=null
SRM-CLIENT: received
TURL=gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/dat
a01/test/2010-05-06/inputfile1.1331.0

SRM-CLIENT: Thu May 06 13:58:21 IST 2010 start file transfer

SRM-CLIENT:
Source=gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2store/storage1/dat
a01/test/2010-05-06/inputfile1.1331.0
srm://xn05/dpm/ctsf.cdac.org.in/home/test/inputfile1
SRM-CLIENT: Thu May 06 13:58:23 IST 2010 Calling releaseFile
SRM-CLIENT:  …Calling srmReleaseFiles…

    status=SRM_SUCCESS
    explanation=null
    status=SRM_SUCCESS
    explanation=null

SRM-CLIENT: Request completed with success
SRM-CLIENT: Printing text report now …
SRM-CLIENT*REQUESTTYPE=get
SRM-CLIENT*TOTALFILES=1
SRM-CLIENT*TOTAL_SUCCESS=1
SRM-CLIENT*TOTAL_FAILED=0
SRM-CLIENT*REQUEST_TOKEN=686b6b39-81bf-497f-a645-65e7f11b69ae
SRM-CLIENT*REQUEST_STATUS=SRM_SUCCESS
SRM-CLIENT*SOURCEURL[0]=srm://xn05/dpm/ctsf.cdac.org.in/home/test/inputfile1
SRM-CLIENT*TARGETURL[0]=file:/tmp/getfile1
SRM-
CLIENT*TRANSFERURL[0]=gsiftp://pvfs4.ctsf.cdac.org.in/pvfs4.ctsf.cdac.org.in:/pvfs2st
ore/storage1/data01/test/2010-05-06/inputfile1.1331.0
SRM-CLIENT*ACTUALSIZE[0]=225
SRM-CLIENT*FILE_STATUS[0]=SRM_FILE_PINNED
```

NOTE : put/get a file to/from GSRM server can be done using same command "srm-copy" in a single step

2.7 BestMan Java APIs Usage :

BestMan Java APIs are available under the samples directory of the client installation directoryCheck the example programs for the basic operations like **ping , ls , get , put , dir** etc under the "samples" directory of " bestman.java.api " folder and add or extend your own functions to perform the required tasks using the available api's from the Bestman java api's document under betman.java.api/docs/javadoc folder .

Table 2 shows the list of Bestman Java classes and the methods supported by them.

Class	Methods
SRMServer SRMServer class is responsible for establishing and maintaining connections with the SRM Server.	connect(java.lang.String serverUrl) disconnect() ping(java.lang.String uid)
SRMRequest The SRMRequest object holds all outstanding request information and status.	*setRequestType*(java.lang.String requestType) *addFiles*(java.lang.String[] surls, java.lang.String[] copyTargetSurls, long[] fileSize) *setAllLevelsRecursive*(boolean allLevelsRecursive) *setAuthID*(java.lang.String authId) *setCount*(int count) *setFileLifeTime*(int fileLifeTime) *setFileStorageType*(java.lang.String tokenType) *setNumberLevels*(int numLevels) *setPinLifeTime*(int pinLifeTime) *setRequestToken*(java.lang.String reqId) *setRetentionPolicy*(java.lang.String retentionPolicy) *setSRMServer*(SRMServer srmServer) *setStorageSystemInfo*(java.lang.String storageInfo) *setTargetSURLLifeTime*(int fileLifeTime) *srmLs()* *srmRm()* *submit()* *checkStatus()* *checkStatus*(java.lang.String[] surl, java.lang.String[] copyTargetSurl) *getAllLevelsRecursive()* *getCount()* *getFiles()* *getFullDetailedList()* *getNumberLevels()* *getRequestSummary()* *getRequestToken()* *getRequestType()* *getSRMServer()* *getStatus()* *printResults()*

	printSummary() *putDone*(java.lang.String[] siteUrl) *releaseFiles*(java.lang.String[] siteUrl) *removeFiles*(java.lang.String[] surls) *abortRequest()*
SRMRequestStatus Show **the status of the**	*getReturnStatus()* *getRequestToken()* *setPathDetails(java.util.HashMappathDetails)* *getPathDetails()* *getFileStatuses()*
SrmRequestReturnStatus	*getStatus()* *getExplanation()*
SRMRequestSummary	*getSummaryArray()*

Table 3 List of Bestman Java classes and APIs

Example 1: A sample Java Program using BestMan Java APIs to Show get a file from GSRM server

Pseudo Code:

```
Initialization ...
.........
  try {
     SRMServer cc = new SRMServer(log4jlocation, logPath, debug);
     cc.connect(serverUrl);
     cc.ping(uid);
     SRMRequest req = new SRMRequest();
     req.setSRMServer(cc);
     req.setAuthID(uid);
     req.setRequestType("get");  //put
     req.addFiles(surl, null,null);
     req.setStorageSystemInfo(storageInfo);
     req.setFileStorageType(fileType);
    req.submit();
   for(int j=0;j<2;j++)
   {
     boolean status;
    status= req.checkStatus();
    System.out.println("req.checkstatus :="+status);
   SRMRequestStatus response = req.getStatus();
     if(response != null) {
      System.out.println("\nStatus.code="+response.getReturnStatus().getStatusCode());
      System.out.println("\nStatus.exp="+response.getReturnStatus().getExplanation());
      if(response.getReturnStatus().getStatusCode() == TStatusCode.SRM_SUCCESS ||
       response.getReturnStatus().getStatusCode()                                    ==
TStatusCode.SRM_FILE_PINNED) {
HashMap map = response.getFileStatuses();
          Set set = map.entrySet();
          Iterator i = set.iterator();
while(i.hasNext()) {
            Map.Entry me = (Map.Entry) i.next();
            String key =  (String) me.getKey();
            Object value = me.getValue();
            if(value != null) {
              FileStatus fileStatus = (FileStatus) value;
```

```
            org.apache.axis.types.URI uri = fileStatus.getTransferSURL();
                System.out.println("\nTransferSURL="+uri);
               turl=uri.toString();
System.out.println("turl is :"+turl);
               String temp[]=turl.split(":2811");

               turl_final=temp[0]+"/"+temp[1];
               System.out.println("final turl is :"+turl_final);
              }
            }//end while
          }//end if
        }//end if
        if(releaseFile) {
          //before release file is called,
                //user will do the file transfer either with globus-url-copy  or with other tool
          //to the SRM server. ie, copy the SRM server's transfer url to the local machine.
          //and when the transfer file is successful, user will call release file.
          req.releaseFiles(surl);
        }
        response = req.getStatus();
        req.printResults();
}
        cc.disconnect();
GlobusURL from = new GlobusURL(turl_final);
 GlobusURL to = new GlobusURL("file:////home/os/pfs/getfile.txt");
System.out.println("##globusurl copy info##");
System.out.println("source url :"+from);
System.out.println("destination url:"+to);
System.out.println("##globusurl copy info######");
    UrlCopy c = new UrlCopy();
    //Setting the urls
    c.setSourceUrl(from);
    c.setDestinationUrl(to);
    c.setUseThirdPartyCopy(true);
    c.copy();
    }catch(Exception e) {
      e.printStackTrace();
    }
   }
```

Execution:

set the classpath ( )

(All jar files which are present under **lib** folder of **bestman root path** should be added to **CLASSPATH**

eg: export CLASSPATH=

/home/test/bestman.java.api/lib/Berkeley.StorageResourceManager-client-api.jar:

.... :

/home/test/bestman.java.api/lib/globus/axis.jar :

.... :

/home/test/bestman.java.api/lib/endorsed/xalan.jar

javac SRMGetTest.java

```
java SRMGetTest  -serviceurl  httpg://xn05.ctsf.cdac.org.in:8446/
                 -surls
srm://xn05.ctsf.cdac.org.in/dpm/ctsf.cdac.org.in/home/garuda/file1
                 -logpath     logfile
                 -desturl     file:///$PWD/getsrmfile
```

Results:

File will be transfered from sourceurl to desturl i.e., from GSRM server to the target location. Check the logs for more information.

Explanation:

In the above code , first the command line parameters are parsed to check for correct values , using those it will connect to the srm server and processes the request based on the request type (eg: get/put) and checks for the request status , once the status returned is SRM_SUCCESS || SRM_FILE_PINNED , we can retrieve the TURL and can be passed to CoG api's to transfer the file from source (turl returned by srm server) to the destination (desturl passed as parameter)

3. GSRM Support

To avail access and support to **GARUDA SRM Storage System,** send a mail to: rt-gds@cdacb.ernet.in

4. Glossary

1. **Storage Element (SE)** : Grid service that allow users to store and manage files on assigned Space.
2. **Logical File Name (LFN)** : alias created by a user to refer to some item of data lfn:/grid/<MyVO>/<MyDirs>/<MyFile>
3. **Site File Name(SFN)**- is a file name assigned by a site to a file. Normally, the site file name will consist of a “machine:port/directory/LFN”, but the site can choose to use another string instead of the LFN.
4. **Grid Unique Identifier (GUID)** : A non-human-readable unique identifier for data entity
5. **Site URL (SURL)** : Physical file name or the location of an actual piece of data on a storage system.<srm>://<SE_hostname>/<path>
6. **Transport URL (TURL)** : Temporary location of a replica +access protocol understood by a SE<protocol>://<path>